Worksheet 5 Bitwise manipulation and masks
Answers

Task 1

1. Figure 1 shows a byte containing a signed integer. The value of the carry bit is unknown.
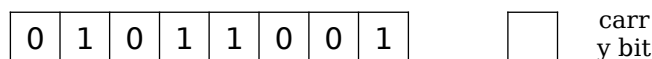
| 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 |

carry bit

Figure 1

Show the result of performing the following shifts, starting each time with the byte given in Figure 1.

(a) a logical right shift:

| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | | 1 |

(b) a logical left shift:

| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | | 0 |

(b) an arithmetic left shift:

| 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | | 1 |

(c) an arithmetic right shift:

| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | | 1 |

2. Using a combination of shifts and addition, multiply 13 by 6

13	0000 1101
13 x 2	0001 1010 (a)
13 x 4	0011 0100 (b)
Add a, b	0100 1110 check 78 = 13 x 6

3. Figure 2 shows an 8-bit byte containing an integer.

| 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | | 0 |

carry bit

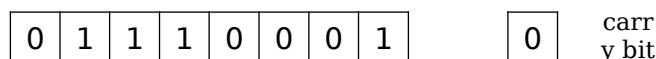*Figure 2*

Show the result of performing the following **consecutive** shifts on the byte.

(a) a circular right shift:

| 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | | 1 |

(b) a circular right shift:

| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | | 0 |

(c) an arithmetic left shift:

| 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | | 0 |

(d) an arithmetic right

1	1	0	1	1	1	0	0

0

shift:

Task 2

4. A system has 8 switches which are controlled by a binary code.Switches 1, 4 and 8 are currently ON.

   (a) It is desired to set switch 7 to a '1' without altering the other switches. Show how this can be done with a mask and a logical operator.

   ```
   Switch number  1 2 3 4 5 6 7 8
   Current state  1 0 0 1 0 0 0 1
   OR with        0 0 0 0 0 0 1 0
   Result         1 0 0 1 0 0 1 1
   ```

   (b) It is now desired to set bits 1 to 4 to 0 without altering bits 5 to 8. Show how this can be done with a mask and a logical operator.

   ```
   Switch number  1 2 3 4 5 6 7 8
   Current state  1 0 0 1 0 0 0 1
   AND with       0 0 0 0 1 1 1 1
   Result         0 0 0 0 0 0 0 1
   ```

5. The ASCII codes for the numbers 0 to 9 are from 0011 0000 to 0011 1001. Using the ASCII code for "1" as an example, show how these can be translated into pure binary using an XOR mask and a logical operator.

   ```
                  1 2 3 4 5 6 7 8
   code for 1     0 0 1 1 0 0 0 1
   XOR with       0 0 1 1 0 0 0 0
   Result         0 0 0 0 0 0 0 1
   ```

   Show an alternative solution using a different mask.

   ```
                  1 2 3 4 5 6 7 8
   code for 1     0 0 1 1 0 0 0 1
   AND with       1 1 0 0 1 1 1 1  (or 0 0 0 0 1 1 1 1)
   Result         0 0 0 0 0 0 0 1
   ```